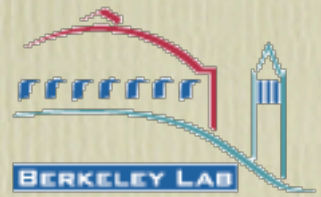

PyLCGDict 2

A Pythonic implementation

Physics Analysis Tools Meeting
11 Feb 2004

Massimo_Marino@lbl.gov

Scripting Services

- RTAG proposes Python to achieve
  - interoperability between CINT and Python
  - scripting and interactive environment
  - component bus
- Python bindings to standard services and utility libraries currently developed in SEAL
  - PyROOT, PyLCGDict, PyBus, further bindings

- Python as a non-intrusive presence in the architecture

Binding Technologies

- RTAG mandate: investigate and recommend
  - various existing options: SWIG, Boost, SIP, raw Python C-API, ...
    - agreed on Boost and Python C-API
- PyLCGDict acting as
  - quick and general solution
  - bridge technology

PyLCGDict

- Python access to C++ libraries (for free)
  - automatic Python proxies to C++ classes using SEAL reflection
    - Pro: hassle-free binding generation
    - Con/Pro: less direct control of interface
  - template, namespace representation
  - Vector-like C++ objects support the Python sequence protocol

PyLCGDict

- Proxies based on information stored in a LCG Dictionary
- Python objects manipulated from C++ via Boost
- First appeared in SEAL 1.0.0, July, C++ based

PyLCGDict 2

- Migrates much of functional core from C++ to Python
- Exploits Python's metaclasses, to make for a flexible, simpler representation
- Better support from Python features, e.g.,
  - iterator protocol
  - introspection features
    - e.g., inheritance tree reflects C++ one
- Python proxies creation loosely coupled to underlying reflection system
  - not constrained by LCG dictionary incarnation

PyLCGDict 2 - status

- Undergoing performance tuning
  - but mostly concentrating on features support
- Extensive test suite of >150 tests
  - write test first than make it work
- On SEAL cvs repository since Monday
  - guinea-pig ready